#4 N17446 US
N17225
item I

Europäisches Patentamt | European Patent Office | Office européen des brevets

jc625 U.S. PTO
09/455662
12/07/99

Bescheinigung | Certificate | Attestation

Die angehefteten Unterlagen stimmen mit der ursprünglich eingereichten Fassung der auf dem nächsten Blatt bezeichneten europäischen Patentanmeldung überein.

The attached documents are exact copies of the European patent application described on the following page, as originally filed.

Les documents fixés à cette attestation sont conformes à la version initialement déposée de la demande de brevet européen spécifiée à la page suivante.

Patentanmeldung Nr. Patent application No. Demande de brevet n°

98204149.3

CERTIFIED COPY OF PRIORITY DOCUMENT

Der Präsident des Europäischen Patentamts;
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

Arlette Fiedler

A. Fiedler

This Page Blank (uspto)

Europäisches Patentamt
European Patent Office
Office européen des brevets

Blatt 2 der Bescheinigung
Sheet 2 of the certificate
Page 2 de l'attestation

Anmeldung Nr.:
Application no.:
Demande n°: 98204149.3

Anmeldetag:
Date of filing: 07/12/98
Date de dépôt:

Anmelder:
Applicant(s):
Demandeur(s):
Koninklijke Philips Electronics N.V.
5621 BA Eindhoven
NETHERLANDS

Bezeichnung der Erfindung:
Title of the invention:
Titre de l'invention:
Enhanced 3D-RS motion estimation algorithm for H.263 video coding

In Anspruch genommene Priorität(en) / Priority(ies) claimed / Priorité(s) revendiquée(s)

Staat: State: Pays:	Tag: Date: Date:	Aktenzeichen: File no. Numéro de dépôt:

Internationale Patentklassifikation:
International Patent classification:
Classification internationale des brevets:
/

Am Anmeldetag benannte Vertragstaaten:
Contracting states designated at date of filing: AT/BE/CH/CY/DE/DK/ES/FI/FR/GB/GR/IE/IT/LI/LU/MC/NL/PT/SE
Etats contractants désignés lors du depôt:

Bemerkungen:
Remarks:
Remarques:

Enhanced 3D–RS Motion Estimation Algorithm for H.263 Video Coding

Stefano Olivieri†

†Philips S.p.A, Philips Research Monza
Via Philips, 12, 20052 Monza (MI), Italy
Tel: +39-039-203 7804 Fax: +39-039-203 7800
E-mail: olivieri@monza.research.philips.com

Abstract

We propose the design of the Enhanced 3D–RS motion estimation algorithm for H.263 video coding, which leads to significant improvement of the coding efficiency with respect to the conventional 3D–RS algorithm, while keeping low the increase of the computational effort. In case of typical videoconferencing sequences the Enhanced 3D–RS algorithm performs well when compared to full–search motion estimation; furthermore, the recursive strategy of the proposed algorithm improves the noise robustness of the estimated motion field, in that its compression gain is comparable to full–search motion estimation in case of noisy sequences. The Enhanced 3D–RS motion estimation algorithm has been successfully integrated with the H.263 video codec for Philips Trimedia processor (TM1000); real–time experiments have proved very good perceptual quality of the coded pictures.

1 Introduction

The H.263 standard for low bitrate video–conferencing [1]–[2] is based on a video compression procedure which exploits the high degree of spatial and temporal correlation in natural video sequences. The hybrid DPCM/DCT coding removes temporal redundancy using inter–frame motion compensation. The residual error images are further processed by block Discrete Cosine Transform (DCT), which reduces spatial redundancy by de–correlating the pixels within a block and concentrating the energy of the block itself into a few low order coefficients. The DCT coefficients are then quantized according to a fixed quantization matrix that is scaled by a Scalar Quantization factor (SQ). Finally, Variable Length Coding (VLC) achieves high encoding efficiency and produces a bitstream, which is transmitted over ISDN (digital) or PSTN (analogue) channels, at constant bitrates. Due to the intrinsic structure of H.263, the final bitstream is produced at variable bitrate, hence it has to be transformed to constant bitrate by the insertion of an output buffer which acts as feedback controller. The buffer controller has to achieve a target bitrate with consistent visual quality, low delay and low complexity. It monitors the amount

of bits produced and dynamically adjusts the quantization parameters, according to its fullness status and to the image complexity.

The H.263 coding standard defines the techniques to be used and the syntax of the bitstream. There are some degrees of freedom in the design of the encoder. The standard puts no constraints about important processing stages such as motion estimation, adaptive scalar quantization, and bit-rate control.

As far as the motion estimation part is concerned, block-matching motion estimation algorithms are usually adopted to estimate the motion field between the current frame to be coded and the previous decoded frame. The objective of motion field estimation for typical hybrid coding schemes is to achieve high motion-compensation performance: however, the evaluation of a large number of candidate vectors for each block can create a huge burden. To save computational effort, a clever search strategy can prevent that all possible vectors need to be checked.

In order to estimate the motion field related to the sequence to be coded, it is possible to use the 3-Dimensional Recursive Search block matching algorithm, presented in [5] and [6]. Unlike the more expensive full-search block matchers that estimate all the possible displacements within a search area, this algorithm only investigates a very limited number of possible displacements. By carefully choosing the candidate vectors, a high performance can be achieved, approaching almost true motion, with a low complexity design.

The 3D-RS algorithm stimulates coherency of the vector field by employing recursion. However, in H.263 video coding context, the extremely smooth estimated motion field impairs the efficiency of the resulting displacement-compensated image prediction. Thus, a compromise must be found between minimizing the entropy of the displacement vectors and minimizing the displaced frame difference between temporally adjacent frames.

For this purpose, we propose the design of the Enhanced 3D-RS motion estimation algorithm which significantly improves the performance in terms of coding efficiency and leads to very good perceptual quality of the coded pictures, while keeping reasonably low the increase of the computational load.

The organization of this document is as follows: Section 2 briefly summarizes the motion estimation part of the video codec, and the design of the 3D-RS motion estimation algorithm is introduced. In Section 3 we describe the architecture of the proposed Enhanced 3D-RS algorithm. In Section 4 the performance in terms of coding efficiency and source distortion is reported. Finally, in Section 5 the conclusions are drawn.

2 Encoding strategy: motion estimation techniques

Motion estimation is part of the inter coding principle. Macroblocks of the current frame are matched to the frame previously coded. In other words, for a specific position, possibly on slightly translated coordinates in the previous frame the best match is found. The underlying necessary translation giving this best match is referred to as the displacement vector. The difference image between the current block and the translated block in the previous frame is referred to as the motion compensated signal. This signal is forwarded to the coding part, in combination with the displacement vector.

2.1 Basic concepts

In block–matching motion estimation algorithms, a displacement vector, or motion vector $\vec{d}(\vec{b}_c, t)$, is assigned to the centre $\vec{b}_c = (x_c, y_c)^{tr}$ of a block of pixels $B(\vec{b}_c)$ in the current image $I(\vec{x}, t)$, where tr means transpose. The assignment is done if $B(\vec{b}_c)$ matches a similar block within a search area $SA(\vec{b}_c)$, also centred at $\vec{b}_c$, but in the previous image $I(\vec{x}, t - T)$. The similar block has a centre which is shifted with respect to $\vec{b}_c$ over the motion vector $\vec{d}(\vec{b}_c, t)$. To find $\vec{d}(\vec{b}_c, t)$, a number of candidate vectors $\vec{C}$ are evaluated applying an error measure $e(\vec{C}, \vec{b}_c, t)$ to quantify block similarity. Figure 1 illustrates the procedure.

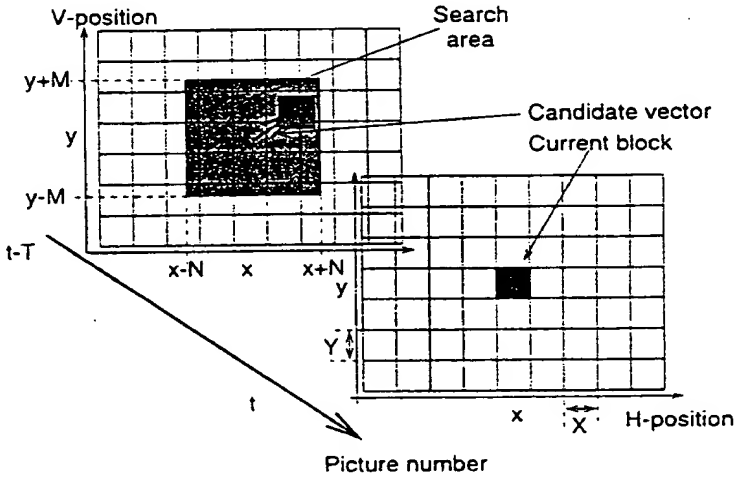


Figure 1: Illustration of block–matching.

The pixels in the block $B(\vec{b}_c)$ have the following positions:

$$(x_c - X/2 \leq x \leq x_c + X/2)$$
$$(y_c - Y/2 \leq y \leq y_c + Y/2)$$

with X and Y the block width and block height respectively, and $\vec{x} = (x, y)^{tr}$ the spatial position in the image.

Although the cost function itself can be rather straightforward and simple to implement, the high repetition factor for this calculation creates a huge burden. This occurs if many

candidate vectors are evaluated, i.e. if large search areas are considered. To save computational effort in block–matching motion estimation algorithms, a clever search strategy has to be designed, preventing that all possible vectors need to be checked.

2.2 3–Dimensional Recursive Search

The high quality 3–Dimensional Recursive Search block matching algorithm, presented in [5] and [6], only investigates a very limited number of possible displacements. By carefully choosing the candidate vectors, a high performance can be achieved, approaching almost true motion, with a low complexity design. Its attractiveness was earlier proven in an IC for SD–TV consumer applications [7].
The 3D–RS algorithm stimulates smoothness of the vector field by employing recursion. In this case the motion field $\vec{\mathrm{d}}(t)$ is given by

$$\vec{\mathrm{d}}(t) \ni \vec{d}(\vec{b}_c, t) = \{\vec{C} \in CS(\vec{b}_c, t) | e(\vec{C}, \vec{b}_c, t) < e(\underline{V}, \vec{b}_c, t))\} \; \forall (\underline{V} \in CS(\vec{b}_c, t)),$$

$$CS(\vec{b}_c, t) = \left\{ \begin{array}{l} (\vec{d}(\vec{b}_c - \begin{pmatrix} X \\ Y \end{pmatrix}, t), \\ (\vec{d}(\vec{b}_c - \begin{pmatrix} -X \\ Y \end{pmatrix}, t), \\ (\vec{d}(\vec{b}_c - \begin{pmatrix} 0 \\ -2Y \end{pmatrix}, t - T)), \\ (\vec{d}(\vec{b}_c - \begin{pmatrix} X \\ 0 \end{pmatrix}, t) + \vec{U}_1, \\ (\vec{d}(\vec{b}_c - \begin{pmatrix} X \\ 0 \end{pmatrix}, t) + \vec{U}_2 \end{array} \right\} \quad (1)$$

$$U_{1x,y} = random(-a_1, \ldots, a_1), \; U_{2x,y} = random(-a_2, \ldots, a_2)$$

where $random(-a, \ldots, a)$ denotes a random choice from the range $[-a, a]$. Figure 2 shows the block diagram of the 3D–RS algorithm.

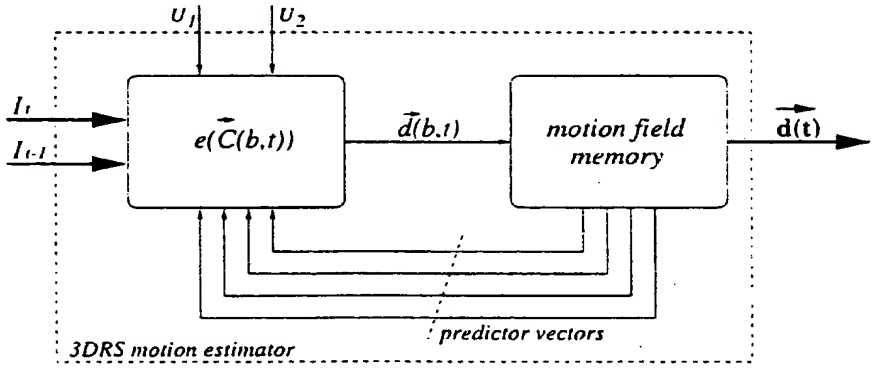


Figure 2: 3D–RS algorithm block diagram.

The candidate set $CS(\vec{b}_c, t)$ consists of 5 vectors: three predictor vectors from a spatio–temporal neighborhood, and two vectors obtained by adding a random update vector to the motion vector estimated for the previous block. This implicitly assumes spatial and/or temporal consistency. Figure 3 shows where the spatial and spatio–temporal prediction vectors are located relative to the current block. In [8] a half–pixel accuracy 3–D Recursive Search block–matcher is proposed, where $[-a_1, a_1] = [-1, 1]$ and $[-a_2, a_2] = [-6, 6]$.
The 3D–RS algorithm leads to extremely smooth vectors fields. This fact reflects its improved coherency strategy (recursive search with spatial and temporal candidates). However, low bitrate H.263 video coding leads to quite poor video quality; moreover, dealing with CIF or QCIF formats, the number of 16 × 16 blocks is relatively small and that causes a slower convergence of the algorithm; these constraints seem to be too strong under certain circumstances, and that makes fall the motion estimator in local minimum errors.

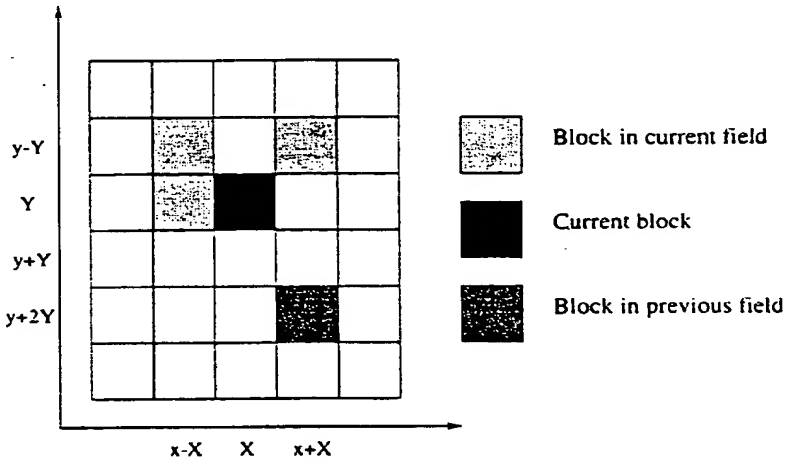


Figure 3: Positions of the prediction vectors relative to the current block.

3 The Enhanced 3D–RS motion estimation algorithm

If we take into account its low computational load (only five displacements are checked), the 3D–RS algorithm is an efficient motion estimator; in [10] is shown a comparison with the full–search motion estimator; for good quality images in the range of 32 to 37 dB PSNR, the average P–frame bitrate increases with only some 5% to achieve the same PSNR. However, in H.263 video coding context the performance of the 3D–RS algorithm is less satisfactory.
Generally speaking, a key for better compression performance is to increase the accuracy of the motion estimator; for example, it is common to use half-pixel accuracy motion estimation in MPEG or H.263 video coding. Instead, in this section we will refer to integer pixel accuracy (without loss of generality) and describe the architecture of the proposed Enhanced 3D–RS algorithm, which improves the performance in terms of coding efficiency while keeping reasonably low the increase of the computational load.

As mentioned in Section 2.2, the recursive strategy of the 3D–RS algorithm stimulates the smoothness of the motion field; this is an advantage because the more the smoothness is, the less the bits spent for motion data are (due to the entropy encoding of the motion information). However, the strong recursion of the 3D–RS algorithm may lead to local minimum errors, which impairs the efficiency of the resulting displacement–compensated image prediction.
Anyway, we can regard the 3D–RS algorithm as a very efficient *coarse* motion estimator, whose estimated motion field needs refining.
In [3],[4] it has been shown that one can exploit the *temporal* recursion of the algorithm by iterating the estimation process several times, in the way that the motion field calculated in the previous iteration is used as temporal candidate field in the current iteration. Therefore, the refinement on the previous estimation is performed *after a vector field.*
A better solution would be to exploit the spatial recursion of the algorithm; if a one–pixel search window refinement around each motion vector at macroblock level is performed, the correction on the currently estimated motion vector is immediately forwarded to the estimation of the next displacement vector.
This solution is shown in Figure 4; the refinement block, inserted into the recursive loop of the estimator, enhances the convergence and speeds up the recursion of the algorithm.
On formulas, the motion field $\vec{\mathbf{d}}(t) \ni \vec{d}(\vec{b}_c, t) = \vec{d}^{\,2}(\vec{b}_c, t)$ is found as[1]

$$\vec{d}^{\,s}(\vec{b}_c, t) = \{\vec{C} \in CS^s(\vec{b}_c, t) | e(\vec{C}, \vec{b}_c, t) < e(\underline{V}, \vec{b}_c, t))\}$$

$$\forall(\underline{V} \in CS^s(\vec{b}_c, t)),\ s = 1, 2$$

$$CS^1(\vec{b}_c, t) = \left\{ \begin{array}{l} (\vec{d}^{\,2}(\vec{b}_c - \begin{pmatrix} X \\ Y \end{pmatrix}, t), \\ (\vec{d}^{\,2}(\vec{b}_c - \begin{pmatrix} -X \\ Y \end{pmatrix}, t), \\ (\vec{d}(\vec{b}_c - \begin{pmatrix} 0 \\ -2Y \end{pmatrix}, t - T)), \\ (\vec{d}^{\,2}(\vec{b}_c - \begin{pmatrix} X \\ 0 \end{pmatrix}, t), \\ (\vec{d}^{\,2}(\vec{b}_c - \begin{pmatrix} X \\ 0 \end{pmatrix}, t) + \vec{U} \end{array} \right\} \qquad (2)$$

$$\vec{U} = (\alpha_x\ \alpha_y) \begin{pmatrix} \mathcal{R}_x \\ \mathcal{R}_y \end{pmatrix},\ \vec{\mathcal{R}} = \left[\vec{d}^{\,2}(\vec{b}_c - \begin{pmatrix} X \\ 0 \end{pmatrix}, t) - \vec{d}^{\,1}(\vec{b}_c - \begin{pmatrix} X \\ 0 \end{pmatrix}, t) \right],$$

$$CS^2(\vec{b}_c, t) = \{\vec{C} | \vec{C} = \vec{d}^{\,1}(\vec{b}_c, t) + \vec{R}\},\ R_{x,y} = \{0, +1, -1\}$$

Equation (2) also shows that a different updating strategy suitable for enhanced estimation can be adopted. No random updates are added to the spatial predictor; this can

[1] concerning the representation of the motion vectors $\vec{d}(\vec{b}_c, t)$ and the candidate sets $CS(\vec{b}_c, t)$, superscript s refers to the step $s = 1, \ldots, N$ in the computation of the motion field $\vec{\mathbf{d}}(t) \ni \vec{d}(\vec{b}_c, t) = \vec{d}^{\,N}(\vec{b}_c, t)$.

be explained by the fact that the refinement process improves the accuracy of the motion estimate; therefore, the displacement vector calculated for the previous block is supposed to be a more reliable predictor to ensure convergence to accurate motion field. In order to enable quick convergence, the update vector $\vec{U}$ is achieved multiplying $\vec{\mathcal{R}}$ by the updating step α, where $\vec{\mathcal{R}}$ is the refinement term related to the previously computed motion vector; in this way the updating process adapts to the local minimum direction. Experimental results proved that the proposed updating strategy leads to some performance improvement with respect to random strategy.

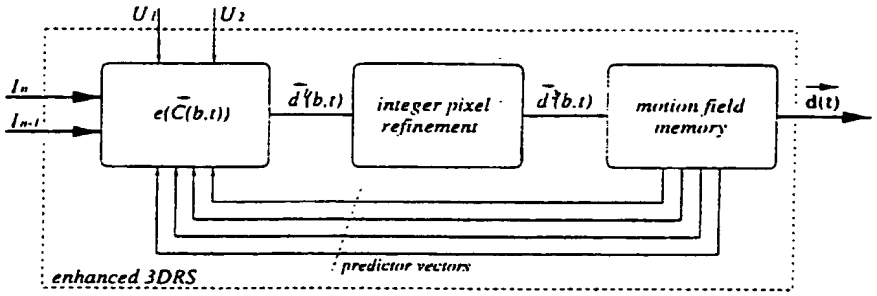


Figure 4: Enhanced 3–Dimensional Recursive search block diagram.

The total number of candidate vectors is 13. Note that, unlike iterative estimation, no additional delay is introduced; indeed, the displacement vectors are immediately available after processing each block.

4 Additional remarks

This invention is not concerned with determining sets of candidate vectors.
In fact I did a lot of experiments about that, and I realized that, as far as H.263 video coding is concerned, one can hardly achieve better performance by simply modifying the set of candidate vectors of the 3D-RS algorithm.

Therefore, I am not proposing to use a particular set of candidate vectors; without loss of generality, I chose that set of candidate vectors according to the version of the 3D-RS algorithm that has been implemented in the IMcIC chip. But I could have chosen any other set of any other version of the 3D-RS algorithm that can be found in the literature; I could also have chosen the candidate vectors of spatially neighboring blocks as candidate vectors, according to EP 0,415,491.

In my opinion, and for what I know, one of the important aspects of this invention is as follows: since the 3D-RS algorithm for H.263 video coding provides quite inaccurate motion vectors, the purpose of the enhancement module is to improve the accuracy of the estimated motion field; thus, the enhancement can be regarded as a post processor of motion vectors. As the enhancement is a post-processing module, it is not involved in determining the set of candidate vectors, as it processes the motion vector associated with the present block, once the best displacement has been selected out of the candidate vectors.

Usually, post-processing is done once the whole motion filed has been computed.

The new aspect of the enhancement of this invention is that better results can be achieved by doing post processing inside the recursion loop of the motion estimation algorithm, provided that the motion field is computed by a recursive motion estimation algorithm. This means that once the motion vector V0 has been selected out of the candidate vectors, said motion vector V0 is refined to produce the motion vector V1, in that if the frame difference corresponding to V1 is smaller than the frame difference associated with V0, V1 immediately replaces V0 before the new set of candidate vectors for the next block is generated (see attached claims for more details).

Note that this technique may be used for any recursive motion estimation algorithm; it can be also used for the motion estimation algorithm described in US 4,853,775, as depicted in the attached block diagram, which shows an inventive improvement over Figure 13 of US 4,853,775.

The post processing of a preferred embodiment of this invention includes an integer pixel refinement around the motion vector that has been selected by the motion estimator; however, any refinement technique able to achieve more accurate motion field may be used. Very good results can be obtained if within the recursion loop, the integer pixel refinement is followed by a half pixel refinement; this solution results, however, in a relatively large computational load, so that the integer pixel refinement is preferred.

Anyway, in my opinion, and for what I know, the new and important aspect is that, provided that post processing is done inside the recursion loop of any recursive motion estimation algorithm instead of outside the recursion loop, the convergence of the recursive motion estimation algorithm is speeded up.

Another important aspect of a preferred embodiment of this invention is that the difference between the output and the input of the enhancement module gives a local information on the trend of the motion (see Equation 2); this information can be exploited to determine an additional candidate vector which contributes to further improve the performance of the algorithm (see also attached block diagram). This can be regarded as an optional feature of the proposed scheme, meaning that the enhancement provides itself conspicuous performance gain, even if the above defined additional candidate vector is not added to the original candidate set.

I would like to remark that both simulation results and subjective tests have confirmed the effectiveness of the enhancement; a lot of time have been spent finding an efficient technique to improve the performance of the 3D-RS algorithm, which provides poor rate-distortion performance when used for H.263 video coding purpose. Although many optimization steps were done to tune various parameters of the 3D-RS algorithm and various techniques to refine the motion field estimated by the 3D-RS algorithm were evaluated, only the adoption of the enhancement scheme described above resulted in considerable performance gain.

5 Conclusion

We have presented the design of the Enhanced 3D-RS motion estimation algorithm for H.263 video coding application, which provides satisfactory performance in terms of coding efficiency while keeping low the computational load. We have shown that the Enhanced 3D-RS algorithm outperforms the improved iterative 3D-RS strategy, and we have found that in case of typical videoconferencing sequences it is very close to full-search motion estimation. Furthermore, we have seen that the recursive estimation strategy stimulates better consistency of the motion field and leads to improved noise robustness of the motion estimation process, in that the Enhanced 3D-RS is comparable with full-search in case of noisy sequences.

The Enhanced 3D-RS algorithm has been successfully integrated with the H.263 video codec for the Philips Trimedia processor (TM1000). It has been seen that this motion estimation algorithm leads to significant computational saving, and real-time experiments have proved very good perceptual quality of the coded sequences.

SDOCID: <E1 9820414903>

References

[1] ITU–T DRAFT Recommendation H.263, Video coding for low bitrate communication, 2 May 1996.

[2] K. Rijkse, "ITU standardisation of very low bitrate video coding algorithms". *Signal Processing: Image Communication* 7 (1995) pp 553–565.

[3] D. Bagni, G. de Haan, L. Albani, C. Alessandretti, V. Riva, "Temporal post–processing for low bitrate video decoders at QCIF resolution", Nat.Lab. Technical Note 317/97.

[4] S. Olivieri, "Enhanced Hybrid Recursive Motion Estimation for H.263 video coding", Nat.Lab. Technical Note 253/98.

[5] G. de Haan, P.W.A.C. Biezen, H. Huijgen, O. A. Ojo, "True motion estimation with 3–D recursive search block matching", *IEEE Trans. Circuits and Systems for Video Technology*, Vol. 3, October 1993, pp. 368–379.

[6] G. de Haan, P.W.A.C. Biezen, "Sub–pixel motion estimation with 3–D recursive search block–matching", *Signal Processing: Image Communication* 6 (1995). pp. 485–498.

[7] P. Lippens, B. De Loore, G. de Haan, P. Eeckhout, H. Huijgen, A. Loning, B. McSweeney, M. Verstraelen, B. Pham, J. Kettenis, "A video signal processor for motion–compensated field–rate upconversion in consumer television", *IEEE Journal of Solid–state Circuits*, Vol. 31, no. 11, November 1996, pp. 1762–1769.

[8] A. van der Werf, R. Kleihorst, E. Waterlander, M. Verstraelen, T. Friedrich, F. Brüls, R. Takken, "I.McIC. A Single–Chip MPEG2 Video Encoder for Storage", Nat.Lab. Report 6979.

[9] E. Barrau, J. Fournier, F. Grolière, "H.263 Video Codec Library Software Description", LEP–Technical Report N. C98–690.

[10] W. Bruls, A. van der Werf, R. Kleihorst, T. Friedrich, E. Salomons, F. Jorritsma, "A single–chip MPEG2 encoder for consumer video storage applications", *ICCE '97*, 1997.

6 Claims

1. A method and an apparatus of improving the accuracy of the motion field estimated by a motion estimation algorithm, which allows improved convergence of the motion estimation algorithm with respect to conventional methods, provided that the motion field is estimated by a recursive motion estimation algorithm, recursive meaning that the motion estimation algorithm computes the motion vector associated with a picture portion (for example a block) by exploiting motion information already determined for previous blocks.

2. A method and an apparatus as claimed in Claim 1 which generates eight motion vectors from the motion vector v_0 that has been selected out of the corresponding candidate vector set. This apparatus is called enhancement module.

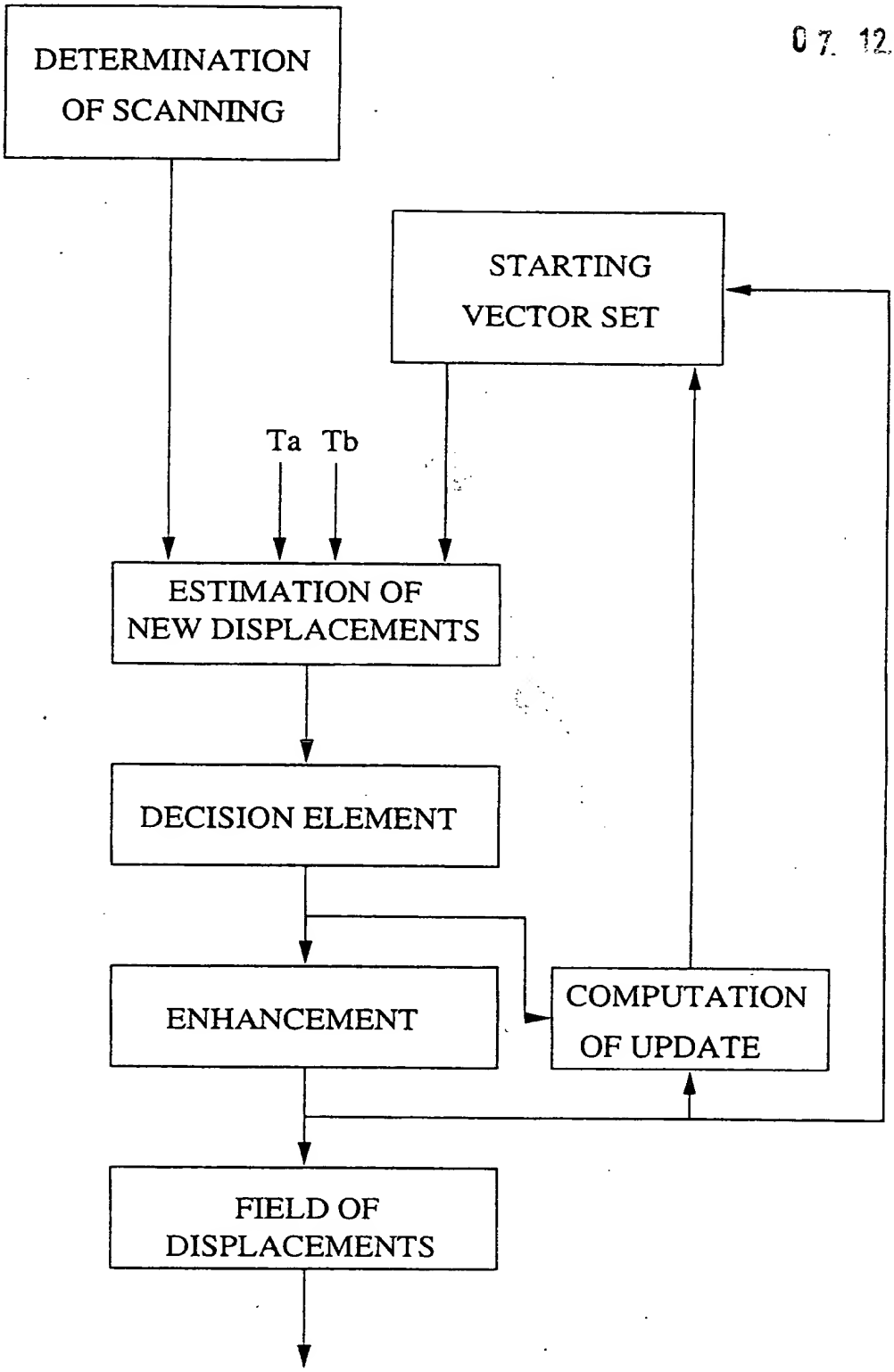
3. A method and an apparatus, according to Claim 2, characterized in that each of the eight vector is achieved by adding ± 1 pixel displacement to each component of the motion vector v_0 that has been selected out of the candidate vector set. The motion vector v_1 out of said eight motion vectors with the smallest frame difference is selected: if the frame difference of said motion vector v_1 is smaller than the frame difference of the motion vector v_0, the motion vector v_1 immediately replaces the motion vector v_0 before the new set of candidate vectors for the motion vector associated with the next block is generated.

4. A method and an apparatus as claimed in Claim 1, which is able to provide a local information on the trend of the motion by computing an update vector given by the difference between output and input of the enhancement. This update vector can be used to generate one more candidate vector by adding the update vector to one of the vectors of the candidate vector set.

5. A method and an apparatus as claimed in any one of the preceding Claims, which speeds up the convergence of the 3D–RS motion estimation algorithm to more accurate motion field.

6. A method and an apparatus as claimed in any one of the preceding Claims, which allows substantial rate–distortion performance gain when applied to H.263 video coding, as well as improved subjective quality.

This page blank (uspto)



BLANK PAGE